Delay Minimization in Varying-Bandwidth Direct Multicast with Side Information

Son Hoang Dau*, Zheng Dong†, Chau Yuen‡
Singapore University of Technology and Design
Emails: {*sonhoang_dau, †dong_zheng, ‡yuenchau}@sutd.edu.sg

Terence H. Chan
Institute for Telecommunications Research
University of South Australia
Email: terence.chan@unisa.edu.au

***Abstract*—We study the delay minimization in a direct multicast communication scheme where a base station wishes to transmit a set of original packets to a group of clients. Each of the clients already has in its cache a subset of the original packets, and requests for all the remaining packets. The base station communicates *directly* with the clients by broadcasting information to them. Assume that bandwidths vary between the station and different clients. We propose a method to *minimize* the *total delay* required for the base station to satisfy requests from all clients.**

I. Introduction

We study the issue of delay minimization of the so-called Direct Multicast with Side Information (DMSI) problem. In an instance of this problem, a base station wishes to transmit a set of n original packets to a group of k clients. Each of the clients already has in its cache a subset of the original packets (referred to as *side information*), and requests for all the remaining packets. The base station communicates *directly* with the clients by broadcasting information to them.

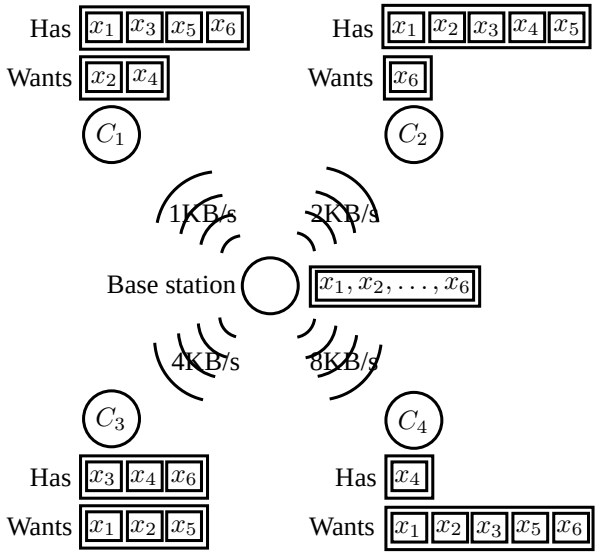

Fig. 1: An example of Direct Multicast with Side Information

Such a scenario is usually observed in opportunistic wireless networks [1], [2], where wireless nodes often opportunistically overhear packets that are not designated to them. These overheard packets become the side information for the nodes. This problem also arises in communication schemes where a server has to broadcast a set of packets to a group of clients. Limited storage capacity, bad reception, or signal degradation might lead to packet loss at the clients. Using a slow feedback channel, the clients inform the server about their missing packets, and request for retransmissions [3].

In our model, each packet that is transmitted from the base station, referred to as a *broadcast* packet, is a linear combination of the *original* packets. Assume that bandwidths *vary* between the base station and different clients, and that each broadcast packet is designated for (in other words, assigned to) *a subgroup* of clients. The *delay* of a broadcast packet is defined to be the amount of time that a client (to which the packet is assigned) with a minimum bandwidth can receive the packet successfully. *Our main contribution* is to provide a method to *minimize* the *total delay* required for the base station to satisfy requests from all clients. We design an optimal packet assignment so as to achieve the minimum total delay. Moreover, the multicast scheme with optimal total delay can be found in polynomial time in n and k.

A motivational example. Suppose that there are four clients C_1, C_2, C_3, C_4, which miss 2, 1, 3, 5 original packets, respectively, as given in Fig 1. By a well-known result in network coding (see Section III for more details), provided that

- the base station broadcasts at least 5 packets to the clients, and
- the number of broadcast packets designated for each client is as many as the number of its missing packets,

then there is a coding scheme for the base station to satisfy demands from all clients simultaneously. Assume that each broadcast packet is of size 8KB and that the bandwidth and the packet delay from each client are given the table in Fig. 2. Note that the delay is obtained by dividing the packet size by the bandwidth between the corresponding client and the base station. Suppose that the base station uses five broadcast

	C_1	C_2	C_3	C_4
Bandwidth (KB/sec)	1	2	4	8
Delay (sec)	8	4	2	1

Fig. 2: Bandwidths and delays for clients

packets p_1,..., p_5. Consider the Packet Assignment A, given in Fig. 3, and the Packet Assignment B, given in Fig. 4. The total delay of the Packet Assignment B (20 seconds) is 4

seconds less than the total delay of the Packet Assignment A (24 seconds). In fact, in Section IV, we can see that Packet Assignment B is actually optimal in terms of the total delay for this scenario. The intuition is that the total delay gets smaller if fewer broadcast packets are assigned to more clients with large delays. This is proved later to be true.

	C_1	C_2	C_3	C_4	Packet delay (sec)
p_1	1		1	1	8
p_2		1		1	4
p_3	1			1	8
p_4			1	1	2
p_5			1	1	2
Total delay					**24**

Fig. 3: Packet Assignment A. A 1-entry means the broadcast packet in that row is assigned to the client in that column. The delay of a broadcast packet is the maximum delay from all clients to which the packet is assigned.

	C_1	C_2	C_3	C_4	Packet delay (sec)
p_1	1	1	1	1	8
p_2	1		1	1	8
p_3			1	1	2
p_4				1	1
p_5				1	1
Total delay					**20**

Fig. 4: Packet Assignment B

Related work. The DMSI problem is a special case of the Multicast with Side Information (MSI) problem [4]. In an MSI instance, there is a network between the base station and the clients. Our problem considers the scenario where the only communication links are those between the base station and the clients. However, the issue of delay minimization is not investigated in [4].

Lun *et al.* [5] study the problem of cost minimization for a general multicast network. In their setting, each vector of rates $\boldsymbol{z}$ at which packets are injected into edges of the network corresponds to a cost $f(\boldsymbol{z})$. The goal is to find $\boldsymbol{z}$ that minimizes $f(\boldsymbol{z})$. The main difference between our result and the result in [5] is the following. The authors in [5] investigate asymptotic solutions with infinite block length codes; in other words, they consider divisible packets with infinitely many subpackets (the non-integral setting). In this work, we are only interested in network codes of block length one; in other words, we only consider indivisible packets (the integral setting). From a practical point of view, solutions to the integral setting are often preferred due to its simplicity in implementation, lower complexity in computation, and smaller buffer required at clients. In general, the integral setting might be harder to tackle than the non-integral setting (for instance linear programming can be solved in polynomial time, whereas integer linear programming is NP-hard). However, in our case, because of the special objective function (the total delay), the optimal solution for the integral setting can be found in polynomial time.

Organization. We formulate our problem rigorously in Section II. A necessary and sufficient condition for the feasibility of a multicast scheme is provided in Section III (Lemma 2). In Section IV, we construct a feasible multicast scheme and prove that it has minimum total delay (Lemma 3, Theorem 4).

II. Problem Definition

A Direct Multicast with Side Information (DMSI) instance is described as follows. A base station S has a set of n *original* packets $X = \{x_1, \ldots, x_n\}$, where $x_i \in \mathbb{F}_q$, $i \in [n]$. There are k clients $C_1, \ldots, C_k$. For each $j \in [k]$, the client C_j possesses a subset of original packets $H_j \subseteq X$ as side information, and demands all missing packets in $X \setminus H_j$. We abbreviate such a DMSI instance by $\mathcal{M} = (n, \{H_j\}_1^k)$.

A *multicast scheme* for the instance $\mathcal{M} = (n, \{H_j\}_1^k)$ is a 2-tuple $(P, \boldsymbol{A})$ where

- $P = \{p_1, \ldots, p_m\}$ is a set of *broadcast packets*, i.e. linear combinations of the original packets, that the base station broadcasts to the clients,
- $\boldsymbol{A} = (a_{i,j})$ is an $m \times k$ binary matrix, where $a_{i,j} = 1$ if and only if the broadcast packet p_i is assigned to the client C_j, for $i \in [m]$, $j \in [k]$.

We refer to $\boldsymbol{A}$ as the (packet) *assignment matrix*. The assignment matrix determines which clients a broadcast packet is assigned to. A multicast scheme is *feasible* if upon receiving all designated broadcast packets, each client can retrieve all missing original packets.

We assume that the client C_j ($j \in [k]$) requires d_j seconds to receipt a broadcast packet (assigned to it) successfully. We refer to d_j as the *delay* from C_j ($j \in [k]$). Furthermore, suppose that after broadcasting a packet $p_i \in P$, the base station can start sending another packet only when all clients that p_i is designated for already receive p_i successfully. We define the *delay* of the packet p_i according to the assignment matrix $\boldsymbol{A}$ by

$$\mathsf{d}_{\boldsymbol{A}}(p_i) = \max\{d_j : \ a_{i,j} = 1\}. \tag{1}$$

Note that the base station must transmit p_i at the minimum rate among all designated clients so that the client with the smallest bandwidth can manage to decode the packet. Therefore, the largest delay among the designated clients is the bottleneck and dominates the delay for that broadcast packet transmission. Therefore, $\mathsf{d}_{\boldsymbol{A}}(p_i)$ is the amount of time required for the broadcast packet p_i to be successfully received by all designated clients. We define the *total delay* of a multicast scheme $(P, \boldsymbol{A})$ by

$$\mathsf{d}_{\sum}(P, \boldsymbol{A}) = \sum_{i=1}^{m} \mathsf{d}_{\boldsymbol{A}}(p_i). \tag{2}$$

Notice that the total delay can be determined solely from the assignment matrix. Therefore, sometimes we use $\mathsf{d}_{\sum}(\boldsymbol{A})$ instead of $\mathsf{d}_{\sum}(P, \boldsymbol{A})$. *Our goal* is to find a feasible multicast scheme with *minimum total delay*, for a given DMSI instance.

As an illustrative example, we consider the DMSI instance as described in Fig. 1. In this example, $n = 6$, $k = 4$, and the side information at the clients are given below.

$$H_1 = \{x_1, x_3, x_5, x_6\},\ H_2 = \{x_1, x_2, x_3, x_4, x_5\},$$
$$H_3 = \{x_3, x_4, x_6\},\ H_4 = \{x_4\}.$$

The Packet Assignment A and B in Fig. 3 and 4 can be incorporated into multicast schemes $(P, \boldsymbol{A})$ and $(P^*, \boldsymbol{A}^*)$, respectively, where the assignment matrices are given in Fig. 5. In Section IV, we show how to determine the packets in P and P^* so that $(P, \boldsymbol{A})$ and $(P^*, \boldsymbol{A}^*)$ are feasible multicast schemes for $\mathcal{M}$. Regarding the packet delay, let us examine

$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \quad \boldsymbol{A}^* = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

(a) (b)

Fig. 5: The assignment matrices $\boldsymbol{A}$ and $\boldsymbol{A}^*$

the third broadcast packet p_3, which is designated for C_3 and C_4, according to $\boldsymbol{A}^*$. The delay from these two clients are $2 = 8/4$ seconds and $1 = 8/8$ seconds, respectively. Therefore,

$$\mathsf{d}_{\boldsymbol{A}^*}(p_3) = \max\{2, 1\} = 2.$$

The total delay of the matrix $\boldsymbol{A}^*$ is calculated as follows.

$$\begin{aligned} \mathsf{d}_{\sum}(\boldsymbol{A}^*) &= \sum_{i=1}^{5} \mathsf{d}_{\boldsymbol{A}^*}(p_i) \\ &= \max\{8, 4, 2, 1\} + \max\{8, 2, 1\} + \max\{2, 1\} \\ &\quad + \max\{1\} + \max\{1\} \\ &= 8 + 8 + 2 + 1 + 1 = 20. \end{aligned}$$

III. Feasibility of a Multicast Scheme via Network Coding

In this section, we establish a necessary and sufficient condition for the feasibility of a multicast scheme for DMSI via network coding.

Hereafter, let $\mathcal{M} = (n, \{H_j\}_1^k)$ be a DMSI instance. For an $m \times k$ binary matrix $\boldsymbol{A}$, we define the network $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ as follows. The set of nodes of $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ consists of

- one source node s, which possesses all original packets $x_1, \ldots, x_n$,
- n "original packet" nodes $s_1, \ldots, s_n$, each corresponds to an original packet,
- m intermediate nodes $u_1, \ldots, u_m$,
- m "broadcast packet" nodes $v_1, \ldots, v_m$,
- k sinks $t_1, t_2, \ldots, t_k$, each corresponds to a client and demands all original packets.

The set of (directed) edges of $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ consists of

- (s, s_i) with capacity one for all $i \in [n]$,
- (s_i, t_j) with capacity infinity if and only if $x_i \in H_j$,
- (s_i, u_h) with capacity infinity for all $i \in [n]$, $h \in [m]$,
- (u_h, v_h) with capacity one for every $h \in [m]$,
- (v_h, t_j) with capacity one if and only if $a_{h,j} = 1$.

As an illustrative example, the network $\mathcal{N}(\mathcal{M}, \boldsymbol{A}^*)$, where $\mathcal{M}$ is given in Fig. 1 and $\boldsymbol{A}^*$ is given in Fig. 5b, is depicted in Fig. 6 and Fig. 7.

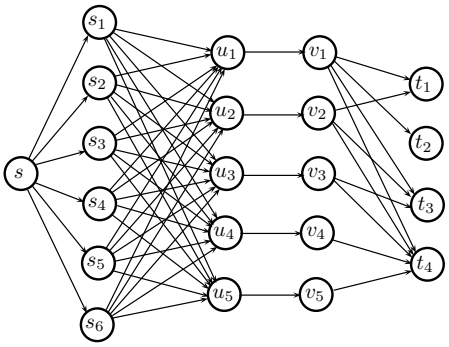

Fig. 6: The network $\mathcal{N}(\mathcal{M}, \boldsymbol{A}^*)$ with $\mathcal{M}$ given in Fig. 1 and $\boldsymbol{A}^*$ given in Fig. 5b. The (side information) edges from s_i to t_j are depicted in a separate figure (Fig. 7) for a clearer view

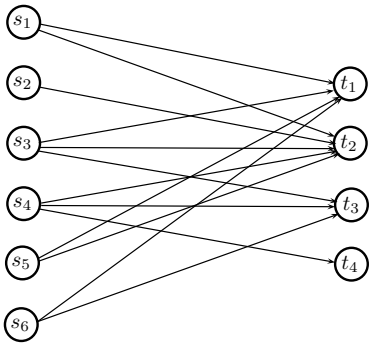

Fig. 7: The side information edges of the network $\mathcal{N}(\mathcal{M}, \boldsymbol{A}^*)$ in Fig. 6

The network $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ is called *solvable* if the source s is able to multicast n packets to all k sinks simultaneously by using a linear coding scheme (see [6]).

Lemma 1. *Suppose that $\boldsymbol{A}$ is an $m \times k$ binary matrix. Then there exists a feasible multicast scheme $(P, \boldsymbol{A})$ with $|P| = m$ for $\mathcal{M}$ if and only if the network $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ is solvable.*

Proof: Assume that there exists a feasible multicast scheme $(P, \boldsymbol{A})$ with $|P| = m$ for $\mathcal{M}$. Let $P = \{p_1, \ldots, p_m\}$. Then s can multicast n packets $x_1, \ldots, x_n$ to all sinks in $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ simultaneously using the following coding scheme:

- s sends x_i to s_i for every $i \in [n]$,
- s_i ($i \in [n]$) sends x_i to t_j ($j \in [k]$) if they are adjacent,
- s_i sends x_i to u_h for every $i \in [n]$ and $h \in [m]$,
- u_h sends p_h to v_h for every $h \in [m]$,
- v_h ($h \in [m]$) sends p_h to t_j ($j \in [k]$) if they are adjacent.

Conversely, assume that the network $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ is solvable. By definition, there is a coding scheme so that s can multicast n packets to all sinks simultaneously. By applying an invertible linear transformation if necessary, we can suppose that s sends x_i to s_i for every $i \in [n]$. For each $h \in [m]$, let p_h be the packet transmitted on the edge (u_h, v_h). Then it is straightforward that $(P, \boldsymbol{A})$ where $P = \{p1, \ldots, p_m\}$ is a feasible multicast scheme for $\mathcal{M}$. ■

For the instance $\mathcal{M} = (n, \{H_j\}_1^k)$, for each $j \in [k]$ let $w_j = n - |H_j|$ denote the number of missing original packets of the client C_j. Let $\mathsf{wt}(\boldsymbol{A}[j])$ denotes the number of 1-entries in the

jth column of $\boldsymbol{A}$. A necessary and sufficient condition for the feasibility of a multicast scheme is presented in the following lemma. We show that it is possible to satisfy demands from all clients if and only if each client receives as many broadcast packets as its missing original packets.

Lemma 2. *Suppose that $\boldsymbol{A}$ is an $m \times k$ binary matrix. Then there exists a feasible multicast scheme $(P, \boldsymbol{A})$ with $|P| = m$ for $\mathcal{M}$ if and only if $\mathsf{wt}(\boldsymbol{A}[j]) \geq w_j$ for every $j \in [k]$.*

Proof: The condition that $\mathsf{wt}(\boldsymbol{A}[j]) \geq w_j$ for every $j \in [k]$ is equivalent to the condition that every cut between the source s and a sink in $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$ has capacity at least n. Due to lack of space, we provide a separate proof for this statement in [7]. By the well-known result from multicast network coding [6], the latter is a necessary and sufficient condition for the solvability of $\mathcal{N}(\mathcal{M}, \boldsymbol{A})$. By Lemma 1, we finish the proof. ∎

Lemma 2 implies that if $(P, \boldsymbol{A})$ is a feasible multicast scheme for $\mathcal{B}$ then $|P| \geq \max_j w_j$. In Section IV-A, we construct a feasible multicast scheme that employs precisely $\max_j w_j$ broadcast packets.

IV. Optimal Packet Assignment

In this section, we first describe a feasible multicast scheme $(P^*, \boldsymbol{A}^*)$ for a DMSI instance $\mathcal{M} = (n, \{H_j\}_1^k)$, and then show that this scheme obtains the minimum total delay among all feasible multicast schemes for $\mathcal{M}$.

A. The Multicast Scheme $(P^*, \boldsymbol{A}^*)$

Relabeling the clients if necessary, we assume that

$$d_1 \geq d_2 \geq \cdots \geq d_k. \tag{3}$$

We consider the multicast scheme $(P^*, \boldsymbol{A}^*)$, where

$$m^* = |P^*| = \max_{j \in [k]} w_j, \tag{4}$$

and $A^* = (a^*_{i,j})$ defined as follows

$$a^*_{i,j} = \begin{cases} 1, & \text{if } 1 \leq i \leq w_j, \\ 0, & \text{if } w_j < i \leq m^*. \end{cases} \tag{5}$$

We already see an example of such an assignment matrix $\boldsymbol{A}^*$ in Fig. 5b, where $\mathcal{M}$ is given in Fig. 1.

The broadcast packets of P^* can be obtained as follows. By (4) and (5), we have $\mathsf{wt}(\boldsymbol{A}[j]) = w_j$ for every $j \in [k]$. Therefore, by Lemma 2, there exist broadcast packets $p_1, \ldots, p_m$ so that $(P^*, \boldsymbol{A}^*)$ with $P^* = \{p_1, \ldots, p_{m^*}\}$ is feasible. Moreover, by the proof of Lemma 1, these broadcast packets can be found in polynomial time in n and k, using the algorithm in [8], given that $q \geq k$. For example, let $q = 4$ and $\mathbb{F}_4 = \{0, 1, \alpha, \alpha^2\}$. For $\boldsymbol{A}^*$ given in Fig. 5b, the broadcast packets of P^* can be chosen as follows: $p_1 = \alpha x_3 + x_4 + \alpha^2 x_5 + \alpha x_6$, $p_2 = x_1 + x_2 + \alpha^2 x_3 + \alpha x_4 + x_5 + x_6$, $p_3 = \alpha x_1 + \alpha^2 x_2 + x_3 + \alpha x_4 + x_5 + \alpha^2 x_6$, $p_4 = x_1 + \alpha^2 x_3 + \alpha x_4 + \alpha^2 x_6$, $p_5 = \alpha^2 x_1 + \alpha x_2 + x_3 + \alpha x_4 + x_5$.

B. The Optimality of $(P^*, \boldsymbol{A}^*)$

Now we prove the optimality of $(P^*, \boldsymbol{A}^*)$ in terms of the total delay. Let $(P, \boldsymbol{A})$ be an arbitrary feasible multicast scheme for $\mathcal{M}$. Our goal is to show that $\mathsf{d}_\sum(\boldsymbol{A}) \geq \mathsf{d}_\sum(\boldsymbol{A}^*)$.

By Lemma 2, the feasibility of $(P, \boldsymbol{A})$ implies that $\mathsf{wt}(\boldsymbol{A}[j]) \geq w_j$ for every $j \in [k]$. Since flipping a 1-entry into a 0-entry does not increase $\mathsf{d}_\sum(\boldsymbol{A})$, we may assume that $\mathsf{wt}(\boldsymbol{A}[j]) = w_j$ for every $j \in [k]$. In Lemma 3, we show that the total delay of $(P, \boldsymbol{A})$ is not smaller than that of $(P^*, \boldsymbol{A}^*)$. First, we illustrate the idea of Lemma 3 via an example.

Consider the DMSI instance $\mathcal{M}$ given in Fig. 1 together with the delays from the clients given in Fig. 2. Let $\boldsymbol{A}$ and $\boldsymbol{A}^*$ be the assignment matrices given in Fig. 5. We now show that $\mathsf{d}_\sum(\boldsymbol{A}) \geq \mathsf{d}_\sum(\boldsymbol{A}^*)$ using an algorithmic approach. We modify $\boldsymbol{A}$ through several steps so that finally, $\boldsymbol{A}$ is turned into $\boldsymbol{A}^*$. Moreover, in every step, $\mathsf{d}_\sum(\boldsymbol{A})$ is never increased.

Step 1. We permute the second and the third row of $\boldsymbol{A}$. Obviously, $\mathsf{d}_\sum(\boldsymbol{A})$ remains unchanged. The matrix now is given in Fig. 8a. We can see that the first columns of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ are now the same.

Step 2. We shift the only 1-entry in the second column of $\boldsymbol{A}$ all the way up to the first row, by swapping $a_{1,2}$ and $a_{3,2}$. The matrix now is given in Fig. 8b. As $d_1 \geq d_2$, the broadcast packet p_1, which corresponds to the first row of $\boldsymbol{A}$, still remains to be d_1 after the aforementioned swap. As $a_{3,2}$ is now zero, the delay of the third packet is decreased to $d_4 \leq d_2$. These are the only changes in the total delay of $\boldsymbol{A}$ after this step. Therefore, $\mathsf{d}_\sum(\boldsymbol{A})$ is not increased (in fact, it is decreased by 3 seconds). Now the first two rows of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ are the same.

Step 3. We first swap $a_{2,3}$ and $a_{4,3}$. The delay of the second broadcast packet is still d_1 after the swap. The delay of the forth broadcast packet, from d_3, is now decreased to $d_4 \leq d_3$. Next, we swap the third and the fifth row of $\boldsymbol{A}$. The total delay of $\boldsymbol{A}$ is unchanged. The matrix now is given in Fig. 8c. The first three rows of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ are the same. Their forth rows are also identical.

$$\begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(a) $\boldsymbol{A}$ after Step 1 (b) $\boldsymbol{A}$ after Step 2 (c) $\boldsymbol{A}$ after Step 3

Fig. 8: $\boldsymbol{A}$ is turned into $\boldsymbol{A}^*$ in three steps

Lemma 3. *Let $\boldsymbol{A}$ be an $m \times k$ binary matrix where $\mathsf{wt}(\boldsymbol{A}[j]) = w_j$ for every $j \in [k]$. Then $\mathsf{d}_\sum(\boldsymbol{A}) \geq \mathsf{d}_\sum(\boldsymbol{A}^*)$.*

Proof: Since $\mathsf{wt}(\boldsymbol{A}[j]) = w_j$ for all $j \in [k]$, we have

$$m \geq \max_{j \in [k]} w_j = m^*.$$

The idea is to repeatedly modify the matrix $\boldsymbol{A}$ through $k+1$ steps, so that at each step, the total delay of $\boldsymbol{A}$ is not increased. At the final step, $\boldsymbol{A}$ is turned into $\boldsymbol{A}^*$. As the total delay never goes up during the whole process, we conclude that $\mathsf{d}_\sum(\boldsymbol{A}) \geq \mathsf{d}_\sum(\boldsymbol{A}^*)$.

Hereafter we say that the two column vectors $\boldsymbol{u} \in \mathbb{F}_q^m$ and $\boldsymbol{v} \in \mathbb{F}_q^{m^*}$ are *almost identical* if their first m^* coordinates are identical and the last $m - m^*$ coordinates of $\boldsymbol{u}$ are all zeros.

Step 1. As $\mathsf{wt}(\boldsymbol{A}) = \mathsf{wt}(\boldsymbol{A}^*) = w_1$, we can permute the rows of $\boldsymbol{A}$ (if necessary) so that the first columns of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ are almost identical. As permuting rows does not affect the total delay, after Step 1, the total delay of $\boldsymbol{A}$ remains the same.

Step j ($2 \le j \le k$). Suppose that up to Step $j-1$, the first $j-1$ columns of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ are almost identical. In this step, we modify $\boldsymbol{A}$ so that the jth columns of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ become almost identical. Intuitively, we shift all of the 1-entries in the jth column of $\boldsymbol{A}$ upward as much as we can, and prove that during the process, the total delay of $\boldsymbol{A}$ is not increased. Let

$$U(j) = \{i \in [m] : \exists j' < j \text{ s.t. } a_{i,j'} = 1\},\ L(j) = [m] \backslash U(j).$$

In words, $U(j)$ denotes the set of upper rows of $\boldsymbol{A}$, each of these contains at least a 1-entry that is located within the first $j-1$ columns. Note that $U(j)$ consists of the first $|U(j)|$ rows of $\boldsymbol{A}$. In opposite, $L(j)$ denotes the set of remaining lower rows of $\boldsymbol{A}$, where all entries in these rows that are located within the first $j-1$ columns are zeros. The following modifications to $\boldsymbol{A}$ do not increase $\mathsf{d}_{\sum}(\boldsymbol{A})$ and at the same time, keep the first $j-1$ columns of $\boldsymbol{A}$ unchanged.

(M1) Modify the entries in the jth column that are located within the first $|U(j)|$ rows. As the first $j-1$ columns of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ are almost identical, the delays (w.r.t $\boldsymbol{A}$) of the first $|U(j)|$ broadcast packets are from the set $\{d_1, \ldots, d_{j-1}\}$. Since $d_j \le d_{j'}$ for all $j' < j$, any change in the jth column within the first $|U(j)|$ rows does not affect the delays of the corresponding packets.

(M2) Turn a 1-entry in the jth column that are located within the last $|L(j)|$ rows into a 0-entry. The delay of the corresponding broadcast packet is changed from d_j to $d_{j''}$ for some $j'' > j$. As $d_{j''} \le d_j$, the packet delay is not increased.

(M3) Permute rows in $L(j)$. It is obvious that permuting rows in $\boldsymbol{A}$ does not affect $\mathsf{d}_{\sum}(\boldsymbol{A})$. Moreover, by definition of $U(j)$ and $L(j)$, permuting rows within $L(j)$ does not affect the first $j-1$ columns of $\boldsymbol{A}$.

With (M1), (M2), and (M3) in mind, we now apply some modifications to $\boldsymbol{A}$. Within the first $|U(j)|$ rows, in the j column of $\boldsymbol{A}$, we swap pairs of 0- and 1-entries such that the 0-entries are below all the 1-entries. Due to (M1), $\mathsf{d}_{\sum}(\boldsymbol{A})$ remains unchanged. We next consider two cases.

(C1) The jth column of $\boldsymbol{A}$ has no 1-entries in the last $|L(j)|$ rows. Then we are done for Step j since now the jth column of $\boldsymbol{A}$ is already almost identical to that of $\boldsymbol{A}^*$.

(C2) The jth column of $\boldsymbol{A}$ has some 1-entries in the last $|L(j)|$ rows. We now examine only the entries in the jth column of $\boldsymbol{A}$.

a) If there are as many 0-entries in the upper part $U(j)$ as 1-entries in the lower part $L(j)$ then we can shift the 1-entries all the way up by applying appropriate entry swaps; and doing so makes the jth columns of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ almost identical. By (M1) and (M2), $\mathsf{d}_{\sum}(\boldsymbol{A})$ is not increased.

b) If there are fewer 0-entries in the upper part $U(j)$ than 1-entries in the lower part $L(j)$, we first shift as many as we can the 1-entries from $L(j)$ to $U(j)$; then all entries in $U(j)$ are one. By (M1) and (M2), $\mathsf{d}_{\sum}(\boldsymbol{A})$ is not increased. Finally, we permute rows in $L(j)$ so that in the jth column of $\boldsymbol{A}$, the 1-entries lie above all the 0-entries. Then the jth columns of $\boldsymbol{A}$ and $\boldsymbol{A}^*$ are almost identical. Moreover, by (M3), $\mathsf{d}_{\sum}(\boldsymbol{A})$ is unchanged.

Step $k+1$. The previous k steps guarantee that $\mathsf{d}_{\sum}(\boldsymbol{A})$ is not increased and all k columns of $\boldsymbol{A}$ are almost identical to that of $\boldsymbol{A}^*$. Therefore, the last $m^* - m$ rows of $\boldsymbol{A}$ are all-zeros. In this step, we remove the last $m^* - m$ rows of $\boldsymbol{A}$, to turn $\boldsymbol{A}$ into $\boldsymbol{A}^*$. Certainly, $\mathsf{d}_{\sum}(\boldsymbol{A})$ remains unchanged in this step. ■

Theorem 4. *The multicast scheme $(P^*, \boldsymbol{A}^*)$ obtains the minimum total delay among all feasible multicast schemes for $\mathcal{M} = (n, \{H_j\}_1^k)$. Moreover,*

$$\mathsf{d}_{\sum}(P^*, \boldsymbol{A}^*) = \sum_{j=1}^{k} d_j \times \max\left\{0, w_j - \max\{w'_j\}_{0 \le j' < j}\right\}, \quad (6)$$

where we set $w_0 = 0$.

Proof: The first assertion follows by Lemma 3. To prove that (6) holds, we show that there are

$$\max\left\{0, w_j - \max\{w'_j\}_{0 \le j' < j}\right\}$$

broadcast packets that have delay d_j. Obviously, the first

$$w_1 = \max\{0, w_1 - \max\{w_0\}\}$$

broadcast packets have delay d_1, due to (1) and (3). By the definition of $(P^*, \boldsymbol{A}^*)$, for each $j > 1$, there are precisely

$$\max\left\{0, w_j - \max\{w'_j\}_{0 \le j' < j}\right\}$$

broadcast packets that are assigned to C_j but to none of the clients $C_{j'}$ with $j' < j$. Due to (1) and (3), these are the only broadcast packets that have delay d_j. ■

V. Acknowledgment

The first author thanks Xiaoli Xu for helpful discussions.

References

[1] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: Practical wireless network coding," in *Proc. ACM SIGCOMM*, 2006, pp. 243–254.

[2] S. Katti, D. Katabi, H. Balakrishnan, and M. Médard, "Symbol-level network coding for wireless mesh networks," *ACM SIGCOMM Comput. Commun. Review*, vol. 38, no. 4, pp. 401–412, 2008.

[3] Y. Birk and T. Kol, "Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2825–2830, 2006.

[4] M. Bakshi and M. Effros, "On achievable rates for multicast in the presence of side information," in *IEEE Int. Symp. Inform. Theory (ISIT)*, 2008, pp. 1661–1665.

[5] D. Lun, N. Ratnakar, M. Medard, R. Koetter, D. Karger, T. Ho, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2608 –2623, 2006.

[6] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, pp. 782–795, 2003.

[7] http://www.sutd.edu.sg/cmsresource/DMSI-ISIT2013.pdf.

[8] S. Jaggi, P. Sanders, P. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inform. Theory*, vol. 51, no. 6, pp. 1973–1982, 2005.